NFVMon

Chirivella Perez, Enrique; Gutiérrez-Aguado, Juan; Alcaraz-Calero, Jose M.; Wang, Qi

Published in:
Wireless Communications and Mobile Computing

DOI:
10.1155/2018/2860452

Published: 14/08/2018

Document Version
Publisher's PDF, also known as Version of record

Hindawi
Wireless Communications and Mobile Computing
Volume 2018, Article ID 2860452, 16 pages
https://doi.org/10.1155/2018/2860452

Research Article

NFVMon: Enabling Multioperator Flow Monitoring in 5G Mobile Edge Computing

Enrique Chirivella-Perez,[1] **Juan Gutiérrez-Aguado**,[2] **Jose M. Alcaraz-Calero**,[1] **and Qi Wang**[1]

[1] *University of the West of Scotland, UK*
[2] *Universitat de Valéncia, Spain*

Correspondence should be addressed to Enrique Chirivella-Perez; enrique.chirivella-perez@uws.ac.uk

Received 23 February 2018; Revised 26 June 2018; Accepted 17 July 2018; Published 14 August 2018

Academic Editor: Chun-Hsian Huang

With the advances of new-generation wireless and mobile communication systems such as the fifth-generation (5G) mobile networks and Internet of Things (IoT) networks, demanding applications such as Ultra-High-Definition video applications is becoming ever popular. These applications require real-time monitoring and processing to meet the mission-critical quality of service requirements and are expected to be supported by the emerging fog or edge computing paradigms. This paper presents *NFVMon*, a novel monitoring architecture to enable flow monitoring capabilities of network traffic in a 5G multioperator mobile edge computing environment. The proposed *NFVMon* is integrated with the management plane of the Cloud Computing. *NFVMon* has been prototyped and a reference implementation is presented. It provides novel capabilities to provide disaggregated metrics related to the different 5G mobile operators sharing infrastructures and also about the different 5G subscribers of each of such mobile operators. Extensive experiments for evaluating the performance of the system have been conducted on a mid-sized infrastructure testbed.

1. Introduction

Advent of fog and edge computing platforms is remarkably shifting the existing communication paradigm by introducing considerable computational, storage and/or networking powers to the edge of the network, especially by leveraging localised Cloud Computing. This trend has become evident particularly in the next-generation mobile networks 5G, where mobile edge computing and cloud radio access networks have been initially standardized recently. Moreover, in 5G, the rapid adoption of hardware virtualization and network encapsulation to provide multitenancy, i.e., to enable multiple 5G operators to share the same physical infrastructure for reducing both capital and operational costs, by virtualizing mobile edge computing infrastructure. This trend in cloudification and softwarization in terms of Network Function Virtualization (NFV) and Software-Defined Networking (SDN), although reducing costs, also brings about the complexity of managing the network. The exploitation of these technologies is indeed reshaping and redefining a significant number of network management concepts that network operators have assumed as valid for years within traditional physical infrastructures and thus pose new challenges to them [1]. Moreover, smart networked systems such as eHealth and smart city applications demand advanced mobile edge computing and networking technologies to provide services to meet challenging quality of service (QoS) requirements such as ultra-low latency and ultra-high reliability for the application traffic flows, among others [2]. In this context, this research work is focused on a fundamental networking capability: monitoring 5G traffic flow metrics in mobile edge computing infrastructures to enable QoS-aware, -informed, or -oriented applications for smart networked systems operating over a 5G mobile edge computing architecture shared by multiple operators for cost-efficiency.

Nowadays, administrative networking perimeters are diluted in multioperator virtual infrastructures that share physical resources. However, traditional hardware monitoring technologies such as sFlow [3], NetFlow [4], IPFix

[5], and OpenFlow Stats [6] are not suitable in this new networking paradigm where the metrics that are collected over a given network interface should be disaggregated for the different operators that share the physical interface. Meanwhile, the software implementation of these technologies, usually supported in the software switches used to enforce operator isolation, does not allow gathering disaggregated metrics. The scenario becomes even more challenging for a 5G mobile network operator, which needs to support user mobility across the different radio access points. It imposes the additional challenge that the level of disaggregated of flow metrics at operator level may not be enough to understand the profile of each of the 5G subscribers of this operator to provide customized QoS-aware services. The solution to resolve these challenges is beyond the state-of-the-art network flow monitoring tools, which has been the main motivation of this research work. It is noted that it is of paramount importance to provide new flow monitoring capabilities at the edge of the network to be able to extract QoS information from multioperator 5G traffic flows in a timely fashion to support delay-sensitive smart system applications and allow better understanding the workload of such 5G infrastructures for operators to manage the network.

The following list summarizes the specific challenges associated with the multioperator virtualized 5G mobile edge infrastructures:

(i) Network traffic generated from virtual machines (VMs) can be seen as disaggregated of the network traffic generated by the network port of the physical machine hosting these VMs. Current flow monitoring tools are not able to provide such disaggregated information, hampering operators to manage the workload of the virtualized infrastructures.

(ii) The life-cycle of VMs introduces new challenges in the monitoring of flow metrics. VMs are created and destroyed dynamically, and the reuse of their IP addresses in other VM occurs in a matter of seconds. Current network flow monitoring tools are not able to detect such reusage, causing inconsistencies in the information gathered about VMs.

(iii) Current network flow monitoring tools do not provide disaggregation of flow metrics at 5G subscriber level with the aim of allowing 5G mobile network operator to gather the profile of each subscriber.

(iv) It is a challenge to achieve timely collection of network metrics without compromising the performance of the data plane. A flow monitoring tool should not impose a noticeable delay in the data path, where user traffic is delivered. Existing flow monitoring tools are typically placed in the middle of the data plane, performing packet classification and metric calculation. This process can cause significant degradation of data plane performance if not implemented in hardware.

This research work has been motivated by the above challenges. The following list provides the main scientific contributions to address these challenges, respectively:

(i) The proposed architecture enables operators to perform the disaggregation by carrying out a customized packet classification that enables inspecting the different network layer encapsulations applied in the infrastructure to allow multioperator isolation through protocols such as VXLAN and GRE.

(ii) The proposed architecture enables detecting the dynamic reusage of IP addresses by integration with the management plane of the mobile edge computing infrastructure.

(iii) The proposed architecture enables them to perform such disaggregation by conducting a customized packet classification of the different network layer encapsulation employed in the infrastructure to handle user mobility through tunneling protocols such as GTP.

(iv) The proposed architecture is based on a monitoring approach where the delay in the data path is not degraded. This is achieved by providing mirroring of a fixed small number of bytes of the packet headers to a tool to undertake such packet header classification and metric calculation tasks.

To the best of our knowledge, this is the first network flow monitoring tool with multioperator capabilities working in a mobile edge computing platform in a 5G network to allow the analyses of the traffic behaviour of each operator and their subscribers in a shared physical infrastructure. The proposed system has been designed, implemented, and deployed using a high-performance approach based on mirroring packets with snap length deployed in OpenVSwitch. Each OpenVSwitch deployed in the architecture has both packet mirroring and snap length enabled. Snap length allows trimming packets to a given length to control the overhead related to monitoring and also to reduce processing times. By doing so, it enables the system to scale at high bitrates while protecting privacy of each user since the payload has been trimmed. From each packet, only the first 140 bytes, mainly the headers of a packet traversing the 5G mobile network infrastructure, are kept instead of the original 1500 bytes defined as the maximum transmission unit (MTU) of the network. This approach reduces up to 90% overhead. An empirical validation using the well-known OpenStack Cloud Computing stack with OpenVSwicth [7], an SDN-enabled software switch used in conjunction with Neutron, the networking services of OpenStack [8], has been carried out. The VNFs used to provide a deployment of an LTE infrastructure with some 5G capabilities are the well-known OpenAirInterface [9].

The rest of the paper is organized as follows. Section 2 describes the state-of-the-art network flow monitoring tools. Section 3 provides a description of the proposed *NFVMon* architecture. Section 4 describes the *NFVMon* application and its integration with the management plane of the network and provides implementation details about the monitoring tool. Section 5 describes the testbed and experiments carried out to validate the prototype. Finally, Section 6 concludes the paper with a discussion and remarks on future work.

Table 1: Comparison of the different monitoring approaches taken in the literature analyzed.

Group ID	Agent Output	Packet Classification	Architecture	Monitoring Collection Protocol
First Group	Packet Metrics	In-Agent	Distributed	Traditional Protocol
Second Group	Packet Metrics	In-Agent	Distributed	SDN Protocol
Third Group	Packet Raw Data	In-Controller	Distributed	Traditional Protocol
Forth Group	Packet Raw Data	In-Controller	Distributed	SDN Protocol

2. Related Work

Despite the recent research interest in mobile edge computing and 5G networks, network flow monitoring tools to fit in these infrastructures are still very scarce. Existing research work on this topic can be grouped by the different approaches taken to achieve flow monitoring. A high-level comparison between the different monitoring approaches analyzed in this section is shown in Table 1.

The first group is composed of the traditional flow monitoring implemented using sFlow [3], NetFlow [4], and IPFIX [5]. They are implemented with a packet classifier to measure metrics about the traditional 5-tuple definition of IP flow (Source IP, Destination IP, Source Port, Destination Port, and Transport Protocol) and report them periodically. This information is received by a Flow Collector for analytical purposes. Mann *et al.* [10] compared sFlow and NetFlow in terms of sampling rate, CPU consumption, and network overhead. The work developed an EMC2 (Edge Monitoring and Collection for Cloud) system that collects and stores metrics from NetFlow and sFlow. Brattstrom et al. [11] propose a scalable agentless monitoring tool for Cloud Computing using SNMP protocol. This proposal deploys a Raspberry Pi device to pull all the monitored devices via SNMP protocol sending the information to a database. Lin et al. [12] propose a monitoring and analytics tool to monitor software-defined network infrastructures. The proposed tools do not provide network administrators with the work load profiles of the infrastructure per mobile operator only counting packets. OmniDisco, proposed by Douitsis et al. [13], is a distributed architecture based on SNMP protocol to monitoring all the devices deployed in the cloud [14]. The paper [15] proposes CloudSuft, a platform-as-a-service solution to monitor networks allocated in different public clouds. To achieve so, different sensors are deployed along the datapath of the different cloud providers. These sensors are used to offer an API that exposes real-time monitoring metrics about the performance of the cloud network. Sminesh et al. [16] propose a flow monitoring scheme to reduce the congestion and packet loss in Software-Defined Networks. The proposal monitors link congestion along the complete path of the flow and then proposes a rerouting of the flow path in order to avoid congested links. Authors directly make use of metrics which imply that packet classification is carried out in the agent. Li et al. propose a method of resource estimation based on QoS in edge computing [17]. Authors classified and matched traffic using a weighted Euclidean distance similarity. The data comes from the monitoring equipment allocated in both data center and edges. Then, monitoring data is sent to the edge servers to analyze the QoS of requested service for end users.

The second group is composed of the flow monitoring tools that follows an SDN controller architecture. This is the natural evolution of the traditional flow monitoring, i.e., first group of tools, by the replacement of the previously mentioned Flow Collector by the SDN Controller. In this context, OpenFlow Statistics extension protocol [18] defined the communication protocols for the sharing of flow metrics between SDN agents and controller. Under this group, it is worth mentioning that OpenNetMon [19], a passive-active flow monitoring tool, implemented for POX SDN controller, allows gathering flow metrics such as throughput, delay, and packet loss. The application polls information about switches at the edge of the network at an adaptive rate to calculate throughput and packet loss. It inserts control packets in the network to measure delay. PayLess [20] is another SDN application for the popular Floodlight SDN [21] controller gathering metrics based on the usage of OpenFlow statistics. In that case, when a *FlowRemoved* packet does not arrive in a time period, a *FlowStatisticsRequest* is sent to the switch, adapting the rate of the next polling interval depending on the variation of the statistics. PayLess can be accessed by a RESTful API. Yoon et al. [22] propose a scalable flow sampling using a centralized approach based on Software-Defined Networks (SDN). This work uses a fixed packet sampling period to capture data packets at switches when an intrusion is detected using an intrusion detection system (IDS). Packet classification and metric calculation are done directly on the switch and metrics are then sent to the traffic analyzer application running in the SDN controller. Yan et al. [23] propose multilayer network analytics architecture based on SDN-based to carry out monitoring of metrics related to the physical layer. These metrics are sent to the SND controller in order to control link impairments, link performance, and latency. An et al. [24] propose a new scheme based on the analysis of the network status in the edge by gathering metrics to extract characteristics and security information of the fog computing infrastructure using Semisupervised Extreme Learning Machines (SS-ELM) algorithm. Hang et al. [25] proposed a solution called Overwatch that provided

a solution to Distributed Denial of Services (DDoS) attacks. Authors developed a collaborative DDoS attack detection mechanism, which consists in a flow monitoring algorithm on the data plane processing the coarse-grained flows and a fine-grained machine learning classification algorithm on the control plane.

The first and second groups of research work have been proved to be effective in physical infrastructures but the fact of using a fix and limited packet classifier hampers users to provide support for the network encapsulation imposed by the usage of multioperator shared infrastructures and the mobility of 5G subscribers across the edges of the network as envisioned in our infrastructure. In fact, it is worthy remarking that the work by Mekky et al. [26] provided a good attempt to extend the packets classifier to analyze more information within packets. They insert logic in the switch by means of app tables allowing gathering metrics using specific application fields.

The third group is composed of the flow monitoring tools that do not perform the packet classification and metric calculation in the same place where the packet is being monitored along the data path between sender and receiver. In this group, Plank [27] used port mirroring to send a copy of the packets to flow collectors. The number of packets sent to the collector is limited by the speed of the connection and the buffer size of that port. The collector calculates the link utilization and flow rates and provides this information to the infrastructure provider. Sharma et al. [28] provide a solution based on hardware switches to allow mobile operators to monitor infrastructures using flexible match+action. This solution is the probably the most optimal in terms of performance but also the most expensive in terms of both operational and capital costs. Our proposed architecture follows a similar approach but is fully implemented in software. Nguyen [29] proposed an analytical method for estimating uncertainty in distribution loads. This method has direct applications to network monitoring and optimal meter placement. The model that author proposes evaluate the impact of aggregated consumption from customers of different types, in order to determine the mean and variance of loading profile in each distribution transformer in the network. Our proposal is looking at the oppose angle where metrics are disgregated in order to achieve fine grain monitoring. Peresini et al. [30] propose a dynamic fine-grained data plane monitoring with Monocle. The probes match the traffic with a specific flow rule in the flow table and push the results to the Software-Defined Networks controller. This architecture is focused on selective monitoring and thus is not suitable to get a global overview of the work load behaviour in the infrastructure.

The fourth group is the natural evolution of the third group by using the SDN controller as the application that performs packet classification, metric calculation, and interface reports. In this group, it is worth highlighting the work by OpenSample [31]. The project uses sFlow[32] algorithm to calculate near real-time metrics of network load and individual flows by using the packets received from the OpenFlow interface, i.e., *Packet_In* messages. FlowSense [33] uses a similar approach where all the *Packet_In* and *FlowRemoved* messages defined in the OpenFlow [6] specification are analyzed; they are associated with flows arrival and flow expiration events. Ze Yang et al. [34] use an SDN network flow monitoring tool to detect "lonely flow first" (LFF). They are flows that only pass by one switch. Our proposal also takes care about this particular use case but using a different name to define such type of traffic, i.e., external traffic. Authors remark that their proposal is highly scalable; however, the real challenge in 5G mobile edge computing is to achieve a monitoring solution to deal with any type of work load. Nguyen-Ngoc et al. [35] propose a solution to perform selective flows monitor (SFM) using ONOS SDN Controller [36]. This SDN application allows classification of flows according to the network administrator requirements. This approach is aligned to our proposed architecture; however, it requires to send the complete packet to the controller and thus suffers of scalability concerns. Lee et al. [37] proposed Duo an intrusion tolerant system SDN, which can reduce exposure time. The solution proposed has two types of servers: long exposure and short exposure time. Then, it classifies the traffic into two different groups, benign and suspicious, that allow dynamical forwarding.

The third and fourth groups provide much more flexibility in the classification of the packet and thus be more flexible to fit in multioperator 5G infrastructures. However, all the research works so far assume that the whole packet is sent to the SDN controller, and thus they tend to double the amount of traffic in the data path. The need of sending a full copy of the packet to the SDN controller obviously raises the concern of scalability issues related to the fact that they will need to deal with the classification of all the packets in every segment of the network being controlled. It is noted that the work by Oliveira *et al.* [38] defined a network flow monitoring tools to capture and process packets to detect network traffic deviations to guarantee the Service Level Agreement (SLA) for each of the operators of a multioperator infrastructure. The work is focused on the description of the overall architecture, although no details are given on the capturing and filtering of packets, the number of tenants or the number of VMs launched in the experiments.

Our proposed architecture was originally enclosed in the fourth group where we created an application for the SDN controller FloodLight to achieve the contributions indicated in this paper. However, we then realized that the current OpenFlow protocol does not allow yet configuring a snap length in order to pass to the controller only a portion of the packet rather than the whole packet to enhances the scalability of the approach and it was a significant effort to implement such capability, which is out of the scope of this research. Thus, it has been decided to provide a design following the approach of the third group, which can be easily ported to the fourth group when such capability becomes available, mainly to deal with scalability by performing packet snapping. It should be noted that this approach is much more optimized to minimize delay in the data path communication between 5G users than the first and second approaches since packet classification and metric calculation are performed outside of the data path. However, the level of responsiveness of the monitoring framework, i.e., the time

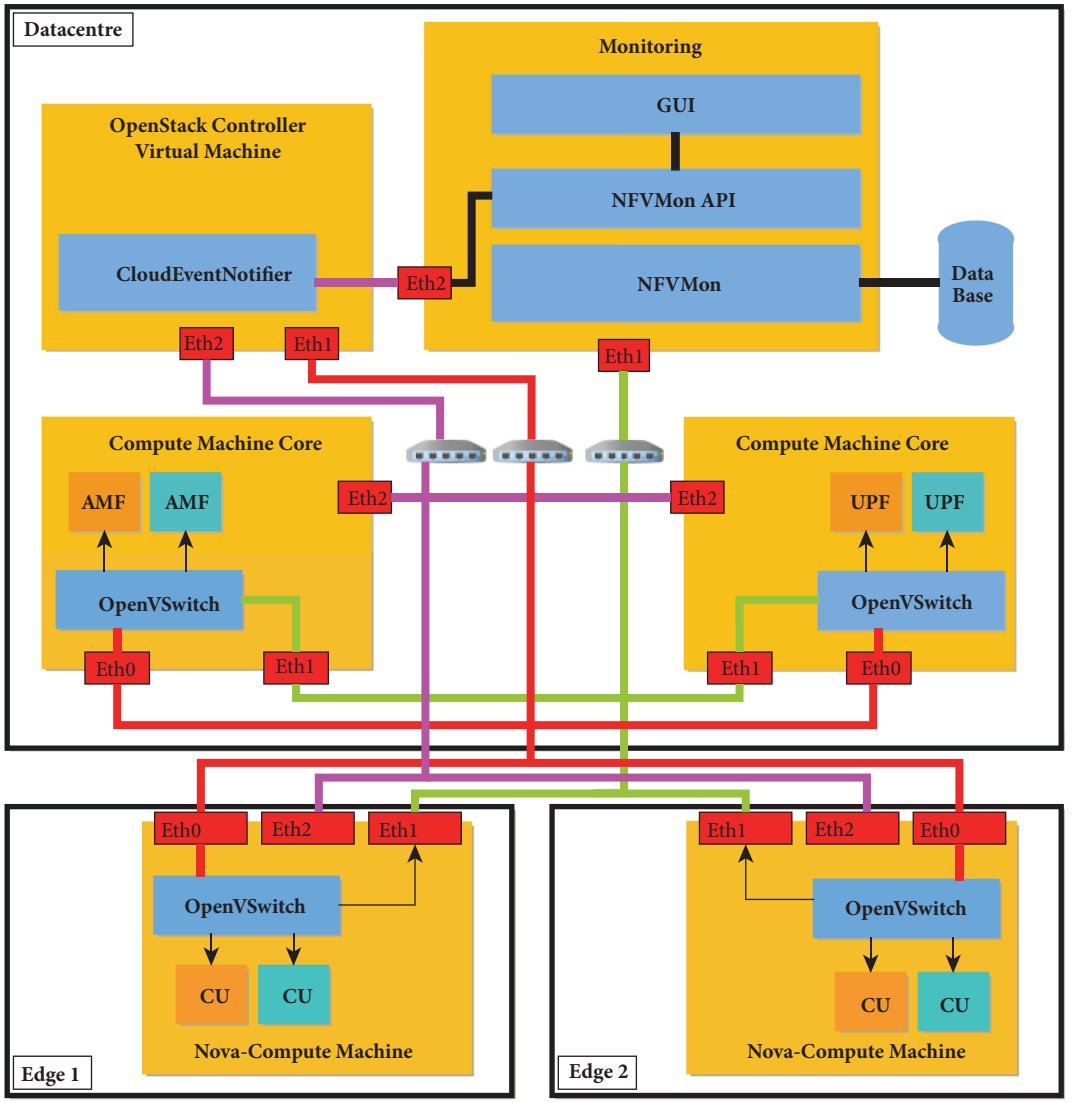


Figure 1: Simplified architecture of *NFVMon*.

elapsed between the moment when the packet is intercepted and when the monitoring framework shows the metrics in screen, will be worse since the framework now needs to deal with packet classification and metric calculation. In terms of scalability and extensibility, our approach also provides better scalability and extensibility architectural principles since it allows scaling up the monitoring framework outside of the real data path and dealing with scalability using an out-of-band approach where the first and second approaches need to consume resources of the machines involved in the data path communications and thus these alternative approaches can lead to performance degradation when addressing high scalability.

3. Proposed *NFVMon* Architecture

Figure 1 shows an overview of the different components available in the proposed *NFVMon* flow monitoring architecture. The architecture is deployed over mobile edge computing infrastructure, where there is a core data center location and several edge locations representing different geographically distributed zones. All locations are managed by the same Cloud Computing infrastructure provider which is in charge of allowing the sharing of physical resources among the different operators involved. To achieve this purpose, the infrastructure provider employs software switches to connect Virtual Machines with the physical network ports while isolating traffic among VMs belonging to different operators. Traffic isolation between different operators is typically achieved by using either tagging (VLAN) or encapsulation protocols (VXLAN, GRE, Geneva). OpenStack supports different types of encapsulation to allow tenant isolation: no encapsulation, VLAN, VXLAN, GRE, etc. VLAN isolates tenants network segments but it does not allow tenants to deploy any networks they need (Software-Defined Networks) on top of the physical network, mainly due to the fact that VLAN relay on the existing MAC addresses of the physical

machines. In contrast, GRE and VXLAN not only provide tenant isolation but also allow any network topology within each of the tenants. This is why these two encapsulation protocols are currently the preferred options for SDN deployments. Moreover, in terms of scalability, GRE and VXLAN provide better scalability when compared with VLAN. VLAN only supports 4096 tenants (12 bits tags), whereas VXLAN and GRE support around 16.7 million of tenants (24 bits tags).

The execution of a software service in an isolated virtualized infrastructure is usually through a Virtual Network Function (VNF). Figure 1 shows two 5G CU (Centralized Unit) VNFs allocated in each edge of the network and two 5G UPF (User Plane Forwarding) VNFs and two AMF (Core Access and Mobility Management Function) allocated in the core of the network. This scenario represents two different 5G mobile operators sharing the same physical infrastructure. Each operator has an isolated virtualized infrastructure where the 5G VNFs are deployed. CU, UPF, and AMF are architectural elements of the novel 5G network representing the control of the radio access network and the control of the user data path and the control of user mobility and authorization, respectively. It is worth noting that the way user mobility is handled in mobile infrastructures is by using an encapsulation protocol to deal with handovers between antennas at the edge of the network, e.g., GTP. A detailed explanation of the whole 5G architecture can be found in Kim et al. [39].

The proposed network flow monitoring architecture is based on mirroring packets passing through the software switches using many-to-one port mirroring with snap length. When packets pass through the software switch, packets are mirrored and snapped at a specific packet length. Traffic snapping is a technique where only the first few bytes of the packet are being mirrored in order to reduce the overhead of the monitoring task. The traffic of an isolated 5G mobile operator running within a shared physical infrastructure in the edge-to-core 5G network segment shows at least a nested encapsulation. The first encapsulation is to achieve tenant isolation (VXLAN/GRE) and the second one is to deal with user mobility within the 5G network (GTP). Thus, the packet headers involved should be the same as or similar to the following header stack: MAC — IP — UDP — VXLAN — MAC — IP — UDP — GTP — IP — UDP/TCP — User Payload. The analysis of these headers allows us to determine the size of the packet header to inform the snapping of network traffic. To be concrete, it is proposed to cut packet header at 140 bytes which is the minimal length of the packets in this network segment. This technique reduces significantly the amount of data mirrored when it is compared with the original packet size. This packet size reduction allows the system to mirror more packets without saturating the monitoring plane, thus fostering scalability. The mirroring enables performing the monitoring tasks outside the user data plane and thus reduces the monitoring overhead in the data path to negligible delay. This approach is very aligned with the strict delay requirements imposed by the 5G infrastructures. Mirroring packet headers allow also the proposed approach to monitor the infrastructure without compromising data privacy of the 5G users due to the fact that any confidential data available in the payload has been removed in the snapping process. In terms of scalability, traffic snapping allows the system to scale up since it only requires around 8-10% of the throughput of the data path (140 bytes out of 1500 bytes).

Regarding the monitoring architectures based on port mirroring outside of the data plane, two main different approaches can be taken. There is no optimal solution and it is a trade-off between accuracy and scalability. Thus, the network administrator according to the use case addressed should choose selective packet monitoring or complete packet monitoring. On the one hand, selective packet monitoring only mirrors key network flows that are defined by the network administrator or any other network management tool. This is used in scenarios where the whole payload needs to be processed and where only a partial status of the network is required, for example, to implement honey nets or other security-related solutions where the complete packet payload should be present in order to be inspected and where only suspicious flows are of interest. Thus, scalability is addressed by mean of the selection of flows. On the other hand, for network monitoring tools whose main aim is to allow network administrator to better understand workloads and to provide a global overview of the network, the complete set of network packets should be handled. This latter use case is in fact the main motivation of this paper. However, it brings significant challenges due to the fact that all the network traffic needs to be processed. Thus, in order to deal with scalability, traffic snapping is proposed to allow the system to reduce traffic in the monitoring plane and processing time in the controller thanks to the fact that it is processing packets of smaller size and as an added value it respects user privacy since packet payload has been trimmed.

This traffic snapping approach allows the architecture to achieve timely network metrics without compromising the performance of the data plane. Moreover, the support of the classifier for the processing of the double encapsulations allows the disaggregation of the flow metrics not only at operator level but also at the 5G subscriber level.

The proposed approach is less scalable than the alternative approach where the packet classification is carried out in its own data path. However, the proposed approach is more optimized for minimizing the delay introduced in the data path due to the classification task carried out therein, specially due to the double encapsulation in 5G multitenanted network. Thus, there is a trade-off between optimization for minimal delay and for maximum scalability. For the mobile edge applications scenario, delay constraint is prioritized.

Snapped traffic is then sent to *NFVMon*. This network Flow Collector receives the snapped traffic, performs packet classification, calculates the interesting metrics, and stores them in a database. In principle, the architecture is flexible enough to provide an extensible set of metrics to be calculated over the snapped traffic. As a mere example, # bytes and # packets per second are traditional metrics that can now be disaggregated at operator level and at the 5G subscriber

level. *NFVMon* exposes an API that allows other modules to interrogate the different metrics to visualize them. The classification makes use of the VXLAN/GRE tunnel ID to allow the disaggregation of metrics associated with the same operator and the IP addresses of the inner encapsulation of the GTP traffic in order to allow the disaggregation of metrics associated with the same 5G subscriber.

There are still issues that need to be addressed in order to achieve a flow monitoring architecture that fits into 5G mobile edge computing architectures. The usage of virtualization imposes a new requirement for operators, i.e., the management of the new life-cycle of virtualized resources, which are different from the traditional physical resources. Now virtual resources can be created and destroyed in a matter of seconds. If the monitoring collector is not aware of these constant topological changes, it will not be able to show to the network administrator the root cause of the current values being reported. To address this issue, a new architectural component has been deployed, i.e., *CloudEventNotifier*.

This component is integrated into the Cloud Computing infrastructure in order to report any topological change happening in the virtual infrastructure. CloudEventNotifier parses all the JSON messages passing through the OpenStack message bus. This change is now received by NFVMon, the proposed network Flow Collector, in order to update the metrics of such affected resources, accordingly. On the other hand, a 5G infrastructure implies mobility of 5G subscribers at the edge of the network. This change is now received into *NFVMon*, our network Flow Collector in order to update the metrics of such affected resources, accordingly. On the other hand, a 5G infrastructure implies mobility of 5G subscribers at the edge of the network.

As a result of this integrated architecture, the infrastructure provider can monitor the quality of the network traffic belonging to different operators that are sharing the infrastructure, and also each of the operators can monitor the traffic flow metrics of their own 5G subscribers.

To explain how the architecture has been integrated, Figure 2 shows a sequence diagram of the orchestration of the different components that compose the *NFVMon* architecture. Thus, *CloudEventNotifier* is registered in the management plane of the infrastructure provider to be informed of any topological change. In this case, a new VM has been notified. When the topological change occurs, the *CloudEventNotifier* interacts with the *NFVMon* Northbound API in order to allow the proper management of the life-cycle of VMs. Meanwhile, there is always a continuous monitoring loop where the snapped traffic is sent to the *NFVMon* Southbound interface. This traffic is then sent to the packet classifier in order to allow the disaggregation of metrics. Once the packet is classified, it is sent to the *NFVMon* Metrics module in charge of the calculation of the metrics and their persistence. Finally, when the infrastructure provider needs to have an overview of the behaviour of all the operators, it requests such information via *NFVMon* Northbound API using the *NFVMon* GUI. These steps are analogous to those to allow each of the operators to view the profile of their 5G subscribers.

4. *NFVMon* Implementation

The integration between the management plane of the infrastructure provider and the data plane of the infrastructure enables *NFVMon* classifying the traffic intercepted in the data plane with a novel categorization of traffic:

(i) Management traffic: traffic exchanged between VMs and *Cloud Controller* services to obtain IP addresses, metadata information, etc.

(ii) Internal traffic: traffic exchanged among VMs within the infrastructure, all the VMs are created by OpenStack, regardless of the geographical locations where they are located.

(iii) External traffic: traffic generated between a VM and a machine that is outside the infrastructure, i.e., the Internet.

Figure 3 depicts an overview of the different aspects of the implementations prototyped. The network traffic is passing through the data plane of the infrastructure. The data plane has OpenVSwitches along the communication among users. Each of the OpenVSwitches is configured to mirror all the packets passing by and trim them with a length of 140 Bytes. When the NFVMon controller receives a new packet, it triggers the packet handler, which stores it directly into a Blocking Queue that contains the trimmed packet with the 5G headers. It is noted that this design allows the packet handler to be really fast and thus minimize packet loss since the processing of the packet is being conducted in separate threads. Then, several classification threads are dequeuing packets from the Blocking Queue in order to perform the packet classification and thus extract the 5G header structure and perform the metric calculation. The headers are used to identify the flow uniquely and as key of two different HashMap concurrent structures. Then the classification threads recalculate the new metrics for the associated flow and perform an update of the values into the global and partial HashMap.

NFVMon keeps updating two main concurrent HashMap data structures: a global structure that keeps statistics along the time and a time-windowed structure that keeps the data produced in a time interval. The time-windowed structure is used to analyze current workload of the infrastructure, whereas the former is more convenient for statistical purposes and predictions. This information is stored in a database for windowed-time visualizations purposes. Each concurrent HashMap structure maintains disaggregated metrics per operator and per 5G subscriber network. Packets in/out and bytes in/out are grouped by different VMs, by different types of traffic (management, internal, and external), by different destination, by source ports, different operators, and by different 5G subscriber. It enables a very detailed monitoring and understanding of the traffic flowing through the infrastructure.

Every N seconds, the updating thread performs the dumping of the content of the HashMap structures into another Blocking Queue, which allows the writing into the database to be conducted in a separate thread and thus does

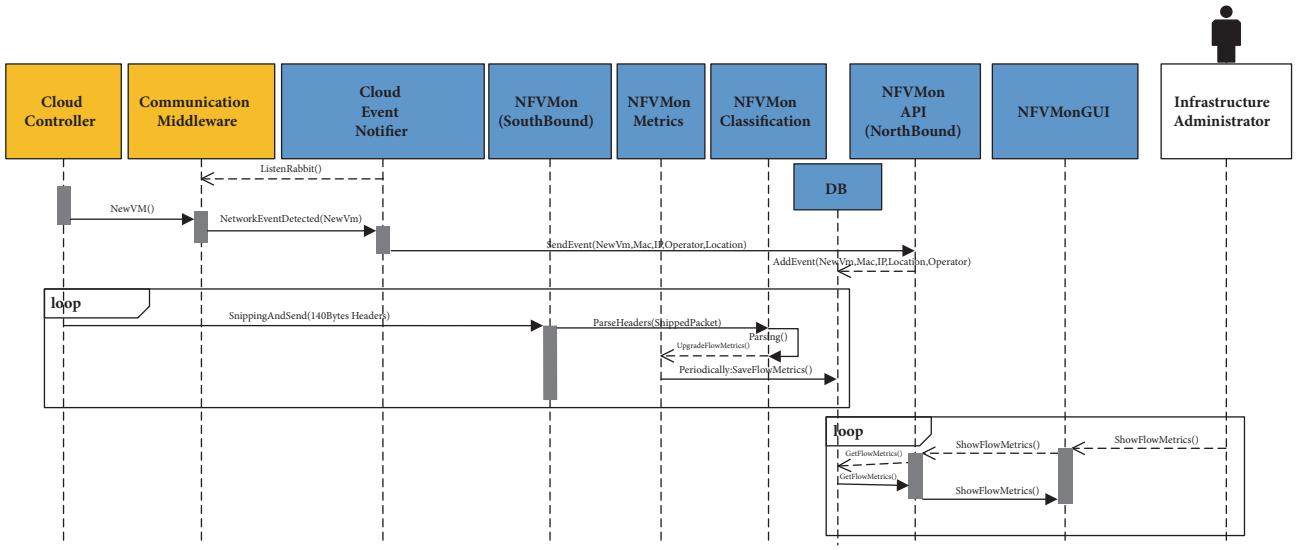


Figure 2: Sequence diagram of the different architectural components available in the proposed architecture.

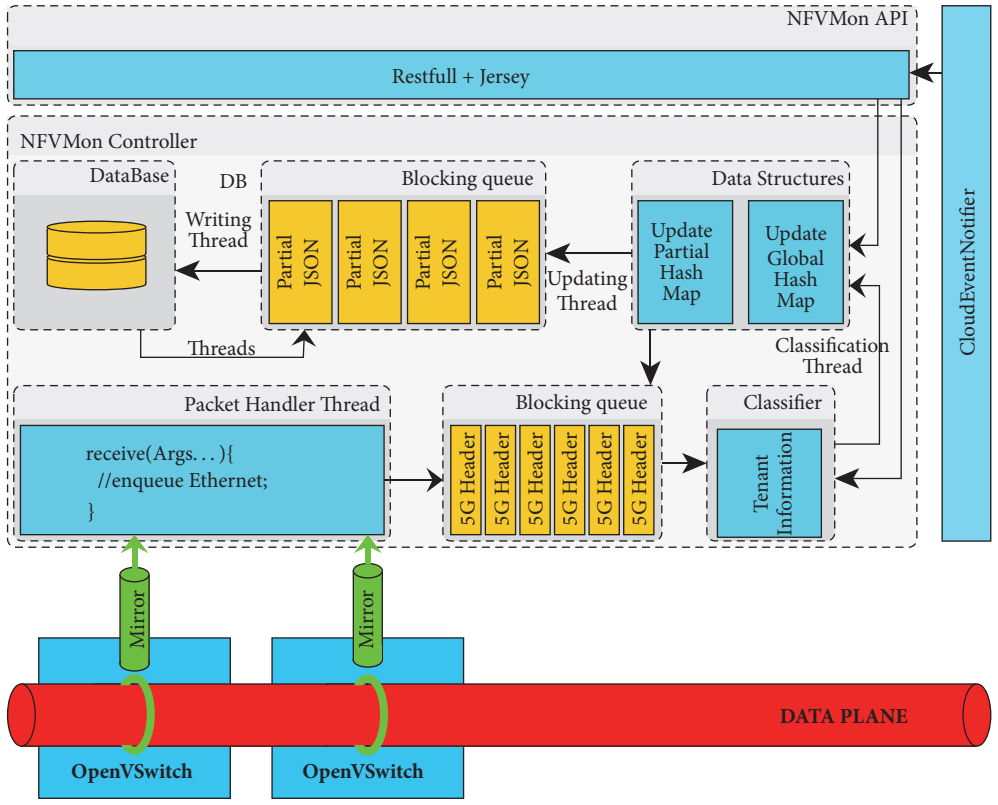


Figure 3: Implementation figure.

not keep locking the HashMap structures to compromise performance. Finally, the Blocking Queue is being consumed by another DB Writing thread that updates the database. The updating thread also cleans the partial HashMap when the information is queued in the Blocking Queue allowing monitoring a time-windowed state of the network. The NFVMon API accesses the HashMap structures when requested by the GUI. The CloudEventNotifier sends changes in the OpenStack topology by using the same NFVMon API. When these changes are received, this information about the tenant and its resources is updated according to the state of the resource reported in the notifications (e.g., terminate VM, remove tenant) and also all the monitoring information stored in the HashMap is updated according to such information.

```
" Operator1 " : {
 "tenantId":"Operator1",
 "VMInUse":25, "VMCreate":25, "VMDelete":0
 ...
 "globalTraffic": {
  "protocolCounter": {
   "17": {
    "protocol":"UDP",
    "idProtocol":17,
    "pktIn":111650,
    "pktOut":111768,
    "appBytesIn":76048464,
    "appBytesOut":76051318,
    "l2BytesIn":80746086,
    "l2BytesOut":81206950},
   "1": { ....}
  }
 },
 "managementTraffic": {
   "03ac4bc8-9158-42e2": {
    "ip":"10.10.0.246",
    "mac":"FA:16:3E:77:CB:1F",
    "vmname":"SGW",
    "destroyDate":0,
    "packetsIn":2991,
    "packetsOut":1825,
    .....
    "proto": {
     "17": {
      "proto":"UDP",
      "id":17,
      ....
      "localPort": {
       "56075": {
        "FA:16:3E:E9:64:FF": {
         "ip":"10.10.0.25",
         "mac":"FA:16:3E:E9:64:FF",
         "packetsIn":4,
         "packetsOut":4,
         ... }
        "FA:13:3E:39:64:FF": {
        .... }
        },
          ....
        }
   }
   }
  "internalTraffic": { ....},
  "externalTraffic": { ....},
 }
```

Listing 1: Example of the JSON data format.

Listing 1 shows a significantly simplified version of the JSON used to store flow metrics in a data structure to allow the reader to understand the disaggregation carried out at different levels. Both global and windowed structures are serialized in JSON format and have a similar structure.

For the use case where an operator needs to delete a VM, the *CloudEventNotifier* receives the messages involved in orchestration and notifies the *CloudEventNotifier* of the information related to the deletion of such VMs. Then, this module renames the associated information to the IP address from both global and time-windowed structures to achieve an efficient handling of the reusage of IP addresses. It will enable differentiating between two different VMs reusing the same IP address.

In terms of implementation details, all the modules developed in this work have been implemented using Java 8. The module *CloudEventNotifier* has been connected to the OpenStack Pike release using OpenVSwitch v2.6 as software switch with a deployment using RabbitMQ as message bus to gather all the messages of the management plane. There is a virtual switch per physical machine where all the VMs allocated to the physical machine are connected.

On the reception of snapped traffic, *NFVMon* has been detached from the processing of packets from the main thread to deal with scalability, by processing the packets in a highly multithread execution context. Thus, the processing of packets in terms of classification and metric calculation does not interrupt the arrival of new packets.

5. Experimental Results

5.1. Validation Testbed. The physical deployment carried out to validate the proposed architecture is based on OpenStack Pike release with Neutron configured with VXLAN tunneling. The infrastructure uses OpenVSwitch *ML2* plug-in for Neutron. It enables the use of OpenVSwitches v2.6 deployed in the infrastructure. The testbed has been deployed in four physical machines with similar specifications: 32 GB RAM, 2 Intel XEON CPU E5-2630 v3 @ 2.40GHz, with 8 cores with hyperthreading, 2 TB Hard Disk, and 10Gb/s network interface cards (NICs). All the physical nodes (machines) have Ubuntu 16.04 Server installed. The architecture deployed is the same as depicted in Figure 1 with the only difference that only one compute has been deployed in the data centre location and such a compute node has also all the components of the OpenStack Controller, including the CloudEventNotifier. The compute nodes and the monitoring node have an additional network interface for the monitoring plane where the snapped traffic is sent. The physical switch used to connect all the computers is a Dell PowerConnect 2824 GbE with three different VLANs in order to isolate the *Management*, *Data Path*, and *Mirrored* networks.

Mobile operators are configured as tenants of the cloud infrastructure, as envisioned in 5G. Each of these operators has deployed four different VNFs. These VNFs correspond to the different architectural elements of the LTE-based 5G networks versus those in the NextGen Core based 5G networks, both of which have been envisioned by 3GPP, which is responsible for standardizing 5G. Specifically, the eNodeB to control radio access in LTE-based 5G is analogous to the DU/CU combination in the NextGen Core based 5G, Mobility Management Entity (MME) analogous to

AMF, Serving and Packet Data Network Gateway (S/PGW) analogous to UPF, and Home Subscriber Service (HSS) analogous to UDM (User Data Management). It is noticed that there is not yet any reference implementation of a NextGen Core based 5G infrastructure ready to deploy the proposed architecture. Therefore, the prototyping of the proposed system has employed the LTE-based 5G network infrastructure based on an advanced open source release of OpenAirInterface [9].

5.2. Validation Results. The effectiveness of the proposed architecture has been validated. To achieve this validation, a testbed has been designed to deploy 12 different mobile operators sharing the same physical infrastructure, each one with a set four VNFS: eNodeB, S/PGW, MME, and HSS. The intention is to deploy always 48 VNFs. Consequently, various scenarios are arranged ranging the number of operators from 1, 2, 4, 8, and 12 and the number of VNFs associated per operators 48, 24, 12, 6, and 4, respectively.

32 User Equipment (UE) nodes are connected to each of the mobile operators. The idea is to fix the number of UE nodes so that they are equally shared between the numbers of eNodeB available in each of the scenarios analyzed.

JMeter has been used to generate HTTP traffic in the UE, VLC to generated RSTP traffic and HTTP Video traffic, and MariaDB to generate database traffic. Each UE only communicates with one remote server for simplicity. The allocation of the VNF for the eNodeB and the S/PGW has been tweaked to make sure they are never allocated in the same physical machine. This deployment forces all the communications to flow between edge and core network segment, thereby avoiding any artificial speedup in the results presented introduced by using loopback interfaces. It is noted that the global traffic generated by the different scenarios is always constant in all the different scenarios analyzed. By arranging so, the amount of traffic generated by each operator is proportional to the number of operators available in each scenario. It means that without any capability of disaggregation of metrics at the operator level, all the scenarios analyzed would show exactly the same workload in traditional flow monitoring tools and no difference between them. However, the proposed architecture allows the infrastructure administrator to have a deep, comprehensive, and complete view of the behaviour of each operator, and the behaviour of the new ongoing network workloads available in the infrastructure. The validation of this new capability is depicted in Figures 4(a) and 4(b).

Figures 4(a) and 4(b) show how the scenarios are completely different despite the fact that they generated exactly the same workloads. The figure shows completely different profiles of each of the operators sharing the infrastructure by using the disaggregated metrics, making a significant differentiating point with respect to the state of the art. It also shows how the numbers of operators causes the sharing of the network traffic among them and the infrastructure administrator is able to understand the traffic behaviour in the infrastructure.

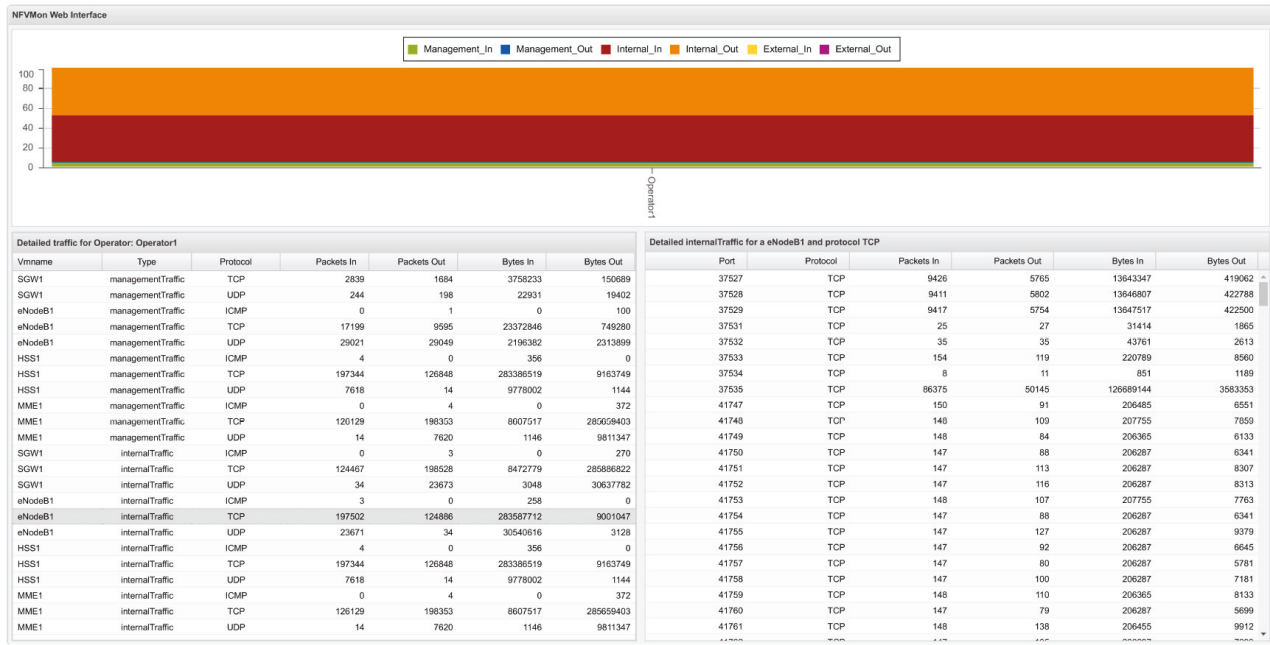
The architecture has been designed to show efficiently and in near real-time flow metrics grouped by operators. All the scenarios have been executed sequentially without shutting down or resetting the proposed architecture. In traditional monitoring tools, without the integration with the management plane of the infrastructure, the metrics associated with the same IP address will be reassigned to another 5G infrastructure during the execution of the next experiment when such infrastructure takes an IP address previously used. This might lead to metrics incorrectly associated with the wrong operator. With our architecture, the testbed is able to deal with the new life-cycle of VMs and thus deleting its associated metrics and dealing with the multioperator scenario thus mitigating these problems.

Figure 5 shows more fine-grained information where the profile of the "Operator 1" shown in Figures 4(a) and 4(b) has been disaggregated in order to allow monitoring the profiling of each of the 32 5G subscribers that are connected in each operator. It is noted that there are 32 different profiles shown in the figure, 16 of which have been configured as clients and another 16 as servers, and this is why there are more or less the same number of bars with the same colour. This behaviour has been concluded due to the disaggregation capabilities provided by the monitoring architecture proposed.

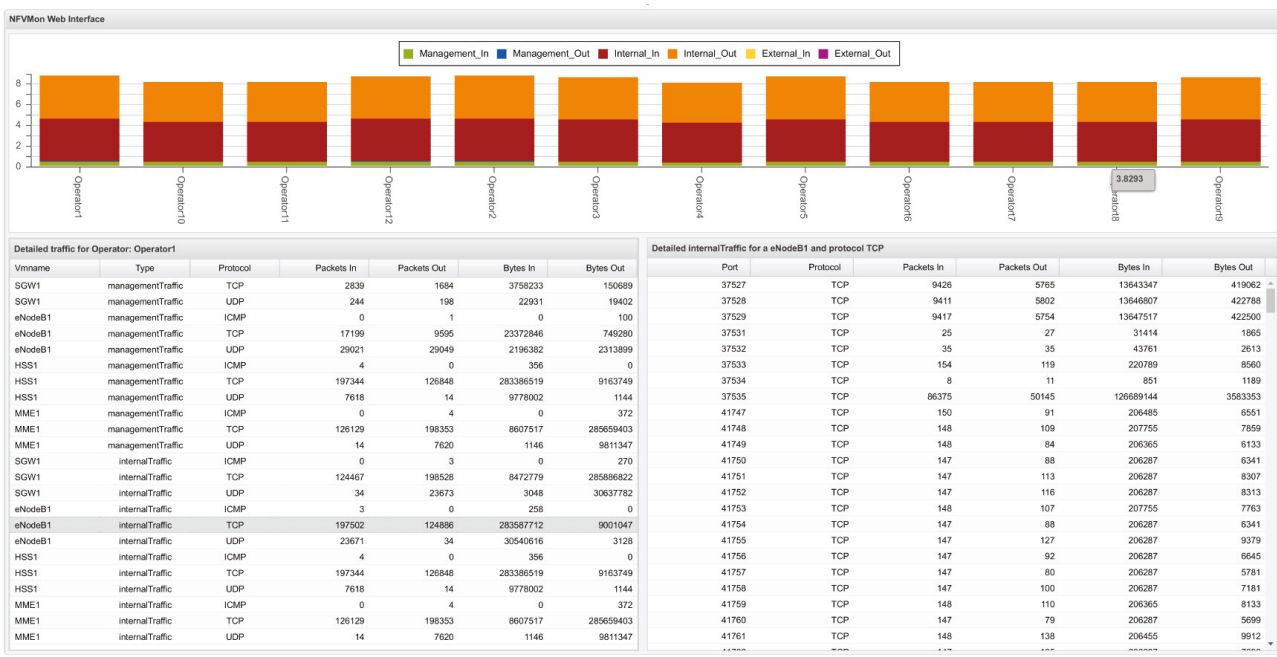
External traffic is not used in any of the experiments for the sake of simplicity. However, the scalability and responsiveness of the prototype will not depend on the type of traffic but on the throughput being processed. And thus, similar results would be expected for any other type of traffic under the similar throughput.

5.3. Performance Results

5.3.1. Delay Results. In order to analyze the overhead of the proposed monitoring system against different workloads, the average delay time between eNodeB and SGW running in a given mobile operator has been measured without our monitoring solution, i.e., no mirror with snapped traffic in the OpenVSwitch and shown in Figure 6. Figure 7 depicts the same scenario but now with our monitoring solutions providing monitoring capabilities and thus the mirrored snapped traffic being enforced in the data path. Comparing both Figures 6 and 7, it can be observed how the activation of the mirrored snapped traffic changes the behaviour in the delay between client and destination in the data path. With mirroring enabled, the delay is much less dependent on the throughput between client and sender and it depends significantly on the packet size. This behaviour is different when compared with the scenario with no mirroring enabled, where both small packet size and higher throughput both contribute to the increase of the delay. In average for all the different traffic workloads analyzed, the overhead delay in the data path by enabling our monitoring solution is 356 microsecond, i.e., 0.356 ms. A closer analysis unveils that this average overhead does not differ significantly between the smallest packet size and the biggest one showing on average 453 and 211 ± 445 microsecond, respectively. The worst case scenario is the less stressed scenario (at 1Gb/s) at the smallest packet size where the data path without mirroring achieved an end-to-end delay of 240 microsecond against



(a)



(b)

Figure 4: Screen shot of the graphical interface for 12 operators, 4 VNFs created.

779 microsecond achieved in the scenario with mirroring (overhead 539 microsecond).

It is noted that even in the worst conditions, usually when a Distributed Denial of Service (DDoS) attack is in place or there is a significant under-provisioning planning in the infrastructure, it can be seen how our proposed network monitoring tool provides a minimal overhead lower than 1 ms; especially when the solution is purely implemented in software, this delay is lower than a millisecond that is the 5G PPP KPI requirement for extremely low latency services. The maximum delay allowed in 5G infrastructures is 10 milliseconds for normal services. Therefore, our proposed solution is 5G compliant [40]. These results presented indicate that *NFVMon* is able to monitor a 5G infrastructure in almost real-time. These results are very reasonable taking into account the fact that all the processing is conducted in

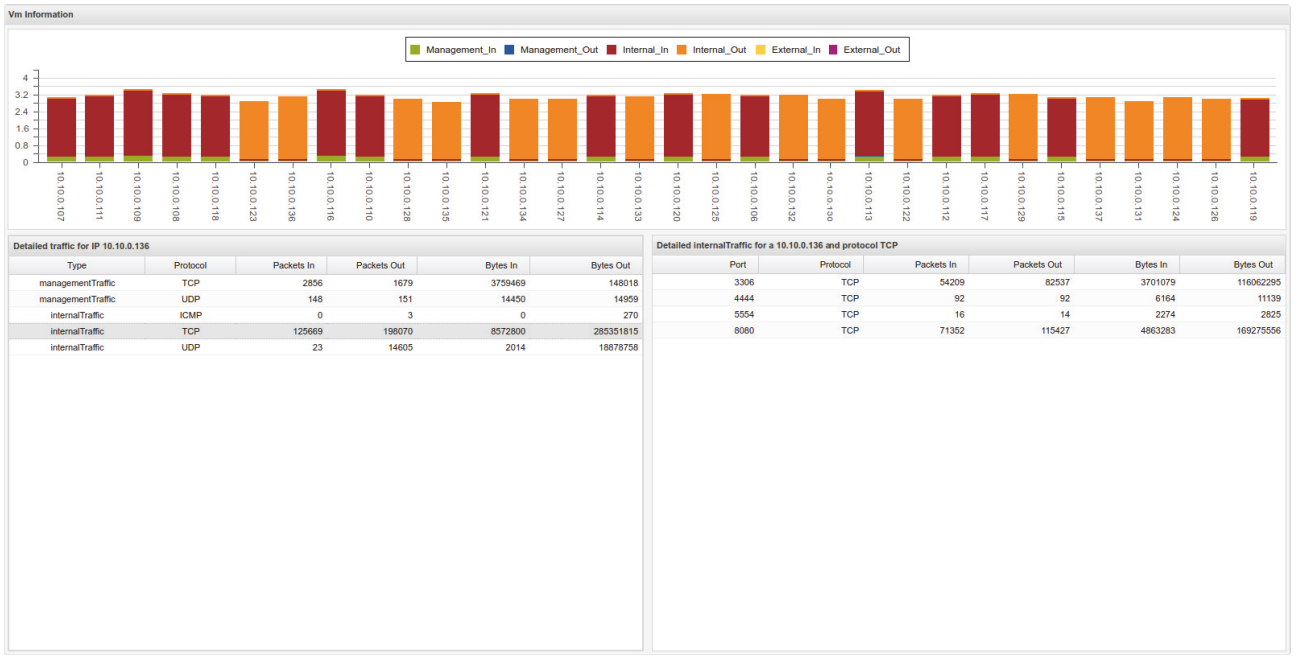


Figure 5: Screen shot of the graphical interface for 48 UEs created for operator 1.

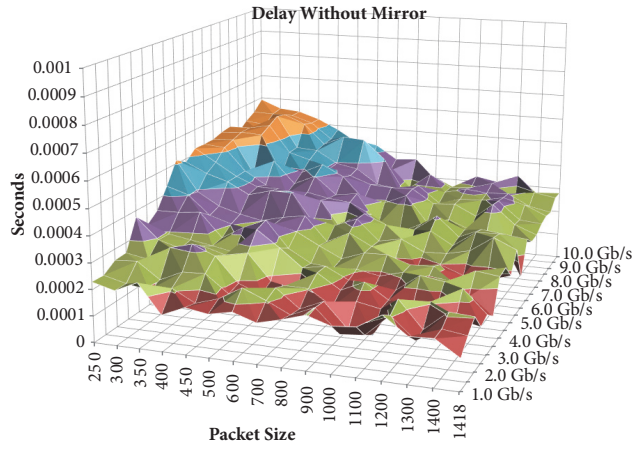


Figure 6: Delay between client and server data path without mirror.

software. There is not any hardware-based acceleration based on FPGA or similar approaches applied in this research, and the application performs deep packet inspection in the payload to be able to extract the information required to carry out the experiments. This architecture is capable of processing all the spectrum with different frame sizes and bandwidth in software. To achieve this high rate, the controller has been optimized to gradually gain efficiency in terms of processing time dedicated to each packet, including multithreading of the *NFVMon*, smart queuing, and adjustment of buffer sizes, among other optimizations.

5.3.2. Scalability Results. Figure 8 depicts the scalability test of the *NFVMon* architecture. It shows how much traffic between client and server (y axis) can be simultaneously monitored by *NFVMon* before starting to lose any packet for a comprehensive range of packet sizes in the communication between client and server (x axis). Packet loss metric can be considered to define the limit of the scalability of the architecture. The different plots available in Figure 9 represent the throughput achieved in the mirrored network (with snapped traffic) before starting to lose packets in the communication to the monitoring VM. As the reader can see, at 400 bytes of packet size onwards, there is a common behaviour for all the scenarios analyzed. In the best case scenario where the packet size is 1418 bytes, *NFVMon* is able to monitor 70Gb/s of traffic between client and server using a 10 Gb/s interface with no packet loss. For 400 bytes of packet size between client and server, the maximum performance to be monitored is around 20 Gb/s. Smaller than this packet size between client and

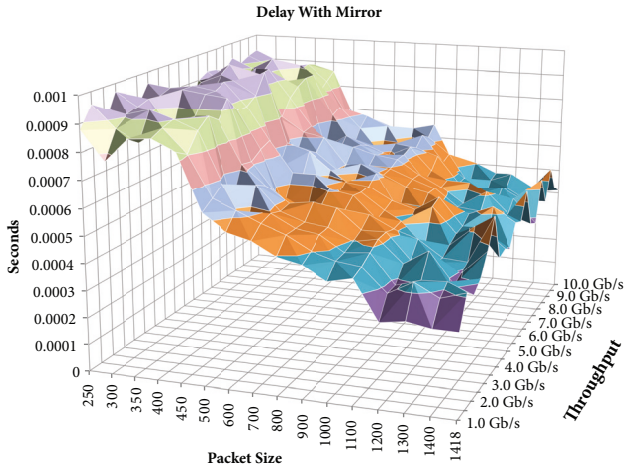


FIGURE 7: Delay between client and server data path with mirror.

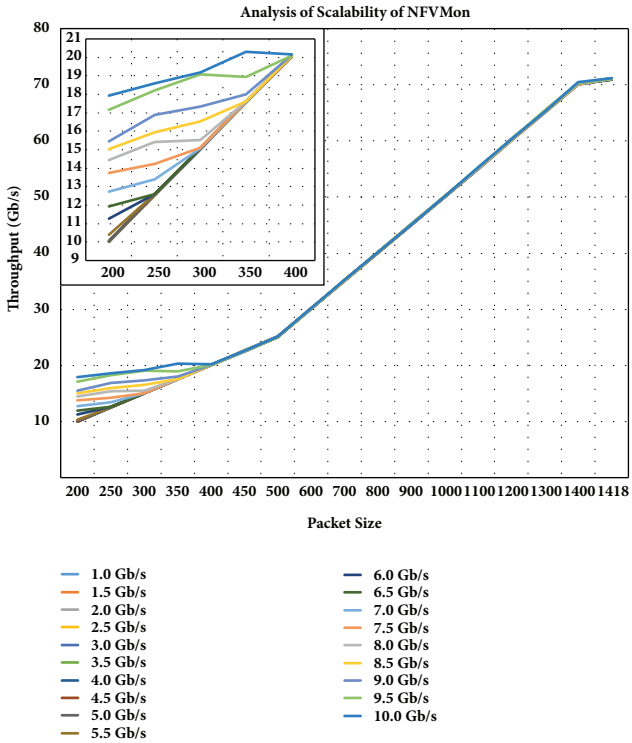


FIGURE 8: Scalability analysis of *NFVMon* architecture proposed measured in the data path between client and monitoring VM.

server, the behaviour is different for the different scenarios but it is due to the fact that the scenario started to resemble a DDoS attack rather than legitimate traffic. All the results presented herein are using a packet snapping length of 200 bytes rather than the originally suggested value of 140 bytes. The reason of this increase is the instrumentation required to carry out the gathering of these results presented herein. It is noted that the measurements require packet sequencing, timestampting, and other instrumentation and it has an impact on the overhead of the packet size. In the production stage, the packet snapping length can be 140 bytes, which

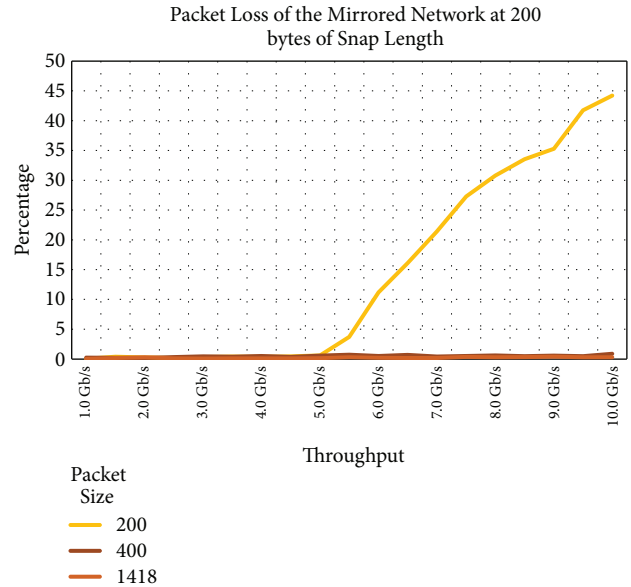


FIGURE 9: Analysis of response time of the flow monitoring tool (delay between client and monitoring tool).

will make these results a bit better in terms of scalability. Therefore, they can be considered as a pessimistic measurement of the scalability of *NFVMon* architecture presented. These results show a significant scalability performance of the proposed solution which mitigates the problems associated with this type of approaches by using a smart traffic snapping technique.

5.3.3. Packet Loss Results. Figure 9 shows the packet loss of the mirrored network according to the stress of this interface. The figure allows the reader to better understand the behaviour of the packet loss for the different key packet sizes analyzed when a snap length of 200 bytes was applied. At both 400 bytes and 1418 bytes of packet size, there is no packet loss happening in the data path of the mirrored network. Then, for the worst case scenario at 200 bytes of packet size, the packet loss started at 5Gb/s reaching up to 45% for 10Gb/s throughput. This scenario is a really stressed scenario that does not represent in somehow real traffic patterns and will be clearly associated with a DDoS attack.

5.3.4. Responsiveness Results. Figure 10 shows an analysis of the responsiveness of the system. It provides the response time when a packet is passing by the data path and it is shown in the GUI interface of the monitoring component. This analysis has considered three main scenarios. The best case is when the packet size is the largest one. In this case, the response time is around 1 ms following a very linear constant trend regardless of the throughput. The worst case scenario, usually under a DDoS attack, is when all the packets are very small. In such a scenario, the response time is 2ms. However, it is worth noting that there is around 40% of packet loss under this circumstance, and the delay is the measurement of the received packets. Then, it has been decided to plot the case

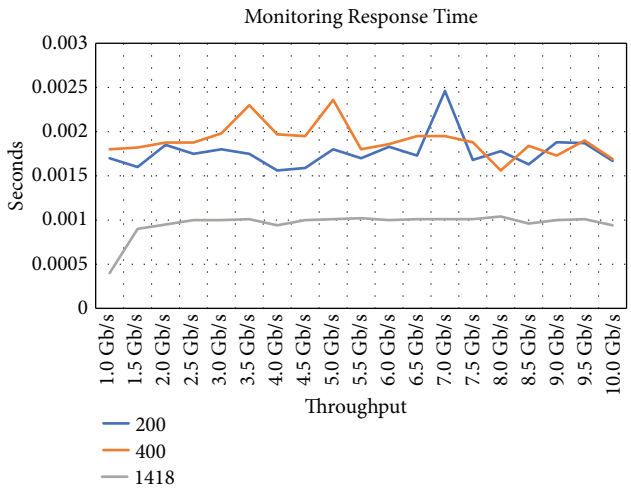


Figure 10: Analysis of response time of the flow monitoring tool.

with 400 bytes of packet size since it is the smallest packet size without packet loss in order to see the response time under this condition. Both cases show a very similar behaviour and all the values follow a linear trend close to 2ms of response time. These values clearly show the resistance and reliability of the proposed system even in very disadvantaged conditions where a DDoS attack is happening in the network.

6. Conclusion and Discussion

The *NFVMon* architecture presented performs efficient and transparent network flow monitoring and analytics of user application traffic at the mobile edge to benefit both QoS-aware smart system applications and multiple 5G operators sharing the same physical infrastructure. The proposed architecture employs a careful integration with the management plane of cloudified infrastructure and SDN/NFV softwarization and visualization technologies. The proposed system has been implemented and validated in a testbed with real 5G cloud infrastructure.

NFVMon enables mobile operators to perform a detailed analysis of network flow monitoring in the cloud-based, SDN/NFV-enabled infrastructure as expected for the emerging 5G infrastructure, and service providers to develop QoS-aware applications for smart networked systems. The proposed system allows analyzing how each VM and each operator is using the infrastructure at a given time and especially how an operator behaves within the cloud infrastructure along the time. Existing solutions for traffic analysis do not classify the traffic per operator or achieve the efficient management of IP reusage due to the lack of integration with the control plane of the cloud infrastructure. The proposed architecture has been intensively tested at 10 Gb/s networks against scalability, overhead delay introduced in the data path, and response time of the monitoring solution. In all the case, the results have demonstrated that this approach can bring a more reliable classification of packets allowing disaggregation of metrics, whereas it can deal also with the scalability problem traditionally associated with this kind of approaches, mainly by the usage of the packet snapping techniques. Results are providing 1-2 ms of response time of the monitoring solution while providing very decent scalability by allowing the monitoring of 20-70 Gb/s in the data path per each of the interfaces being used into the mirrored network. These results are very promising specially when the solution provided is a pure software-based implementation with no hardware-acceleration capabilities.

Future work is expected to explore the usage of the tenant-aware information and network classification as the foundation to explore the provisioning of tenant-aware QoS within modern mobile edge computing infrastructures for 5G communications. It is also envisioned to explore future testbed with higher bitrates such as 100 Gb/s and 400 Gb/s in order to determine the suitability of this approach under such conditions.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was funded in part by the European Commission Horizon 2020 5G-PPP Program under Grant Agreement no. H2020-ICT-2016-2/761913 (SliceNet: End-to-End Cognitive Network Slicing and Slice Management Framework in Visualized Multi-Domain, Multitenant 5G Networks) and by the European Commission Horizon 2020 5G-PPP Programme under Grant Agreement no. H2020-ICT-2014-2/671672-SELFNET (Framework for Self-Organized Network Management in Virtualized and Software-Defined Networks).

References

[1] 5G Infrastructure PPP Association and others, 5G Vision-The 5G Infrastructure Public Private Partnership: the next generation of communication networks and services, White Paper, February 2015.

[2] D. P. Agrawal, B. B. Gupta, S. Yamaguchi, and K. E. Psannis, "Recent advances in mobile cloud computing," *Wireless Communications and Mobile Computing*, vol. 2018, Article ID 5895817, 1 page, 2018.

[3] P. Phaal, S. Panchen, and N. McKee, "InMon corporation's sFlow: A method for monitoring traffic in switched and routed networks," Tech. Rep., 2001.

[4] B. Claise, *Cisco systems netflow services export version 9*, 2004.

[5] B. Claise, "Specification of the IP Flow Information Export (IPFIX) Protocol for the Exchange of IP Traffic Flow Information," RFC Editor RFC5101, 2008.

[6] N. McKeown, T. Anderson, H. Balakrishnan et al., "OpenFlow: enabling innovation in campus networks," *Computer Communication Review*, vol. 38, no. 2, pp. 69–74, 2008.

[7] B. Pfaff, J. Pettit, T. Koponen, E. J. Jackson, A. Zhou, J. Rajahalme et al., "The design and implementation of OpenVSwitch," in *12th USENIX Symposium on Networked Systems Design and Implementation*, vol. 1, pp. 1–14, USENIX, Oakland, CA, USA, 2015.

[8] R. Kamboj and A. Arya, "OpenStack: open source cloud computing IaaS platform," *International Journal of Advanced Research in Computer Science and Software Engineering*, vol. 4, no. 5, pp. 1200–1202, 2014.

[9] N. Nikaein, E. Schiller, R. Favraud et al., "Network store: exploring slicing in future 5G networks," in *Proceedings of the 10th International Workshop on Mobility in the Evolving Internet Architecture (MobiArch '15)*, pp. 8–13, ACM, New York, NY, USA, September 2015.

[10] V. Mann, A. Vishnoi, and S. Bidkar, "Living on the edge: monitoring network flows at the edge in cloud data centers," in *Proceedings of the 5th International Conference on Communication Systems and Networks (COMSNETS '13)*, pp. 1–9, IEEE, Bangalore, India, January 2013.

[11] M. Brattstrom and P. Morreale, "Scalable Agentless Cloud Network Monitoring," in *Proceedings of the 44th International Conference on Cyber Security and Cloud Computing (CSCloud)*, pp. 171–176, IEEE, June 2017.

[12] J. Lin, Q. Zhang, B. Park, H. Bannazadeh, and A. Leon-Garcia, "MonArch: Monitoring and analytics in software defined infrastructures," in *Proceedings of the 4th IEEE International Conference on Cloud Networking, CloudNet 2015*, pp. 198–200, IEEE, October 2015.

[13] A. Douitsis and V. Maglaris, "Towards a scalable management collector," in *Proceedings of the 2016 Global Information Infrastructure and Networking Symposium, GIIS 2016*, pp. 1–6, IEEE, October 2016.

[14] J. Case, M. Fedor, M. Schoffstall, and J. Davin, "Simple Network Management Protocol (SNMP)," RFC Editor RFC1157, 1990.

[15] V. Persico, A. Montieri, and A. Pescape, "CloudSurf: A platform for monitoring public-cloud networks," in *Proceedings of the 2nd IEEE International Forum on Research and Technologies for Society and Industry Leveraging a Better Tomorrow, RTSI 2016*, pp. 1–6, IEEE, September 2016.

[16] C. N. Sminesh, E. G. M. Kanaga, and K. Ranjitha, "Flow monitoring scheme for reducing congestion and packet loss in software defined networks," in *Proceedings of the 4th International Conference on Advanced Computing and Communication Systems, ICACCS 2017*, pp. 1–5, January 2017.

[17] G. Li, J. Song, J. Wu, and J. Wang, "Method of resource estimation based on qos in edge computing," *Wireless Communications and Mobile Computing*, vol. 2018, Article ID 7308913, 9 pages, 2018.

[18] Open Networking Foundation, OpenFlow Switch Specification, Version 1.5.1, October 2015.

[19] N. L. M. Van Adrichem, C. Doerr, and F. A. Kuipers, "OpenNetMon: network monitoring in openflow software-defined networks," in *Proceedings of the IEEE/IFIP Network Operations and Management Symposium: Management in a Software Defined World, NOMS 2014*, pp. 1–8, IEEE, May 2014.

[20] S. R. Chowdhury, M. F. Bari, R. Ahmed, and R. Boutaba, "PayLess: A low cost network monitoring framework for software defined networks," in *Proceedings of the IEEE/IFIP Network Operations and Management Symposium: Management in a Software Defined World, NOMS 2014*, pp. 1–9, IEEE, May 2014.

[21] Indigo, "Floodlight SDN controller," 2018, http://www.projectfloodlight.org/indigo/.

[22] S. Yoon, T. Ha, S. Kim, and H. Lim, "Scalable traffic sampling using centrality measure on software-defined networks," *IEEE Communications Magazine*, vol. 55, no. 7, pp. 43–49, 2017.

[23] S. Yan, A. Aguado, Y. Ou, R. Wang, R. Nejabati, and D. Simeonidou, "Multilayer network analytics with SDN-based monitoring framework," *Journal of Optical Communications and Networking*, vol. 9, no. 2, pp. A271–A279, 2017.

[24] X. An, X. Zhou, X. Lü, F. Lin, and L. Yang, "Sample selected extreme learning machine based intrusion detection in fog computing and MEC," *Wireless Communications and Mobile Computing*, vol. 2018, Article ID 7472095, 10 pages, 2018.

[25] B. Han, X. Yang, Z. Sun, J. Huang, and J. Su, "OverWatch: a cross-plane ddos attack defense framework with collaborative intelligence in SDN," *Security and Communication Networks*, vol. 2018, Article ID 9649643, 15 pages, 2018.

[26] H. Mekky, F. Hao, S. Mukherjee, Z.-L. Zhang, and T. V. Lakshman, "Application-aware data plane processing in SDN," in *Proceedings of the 3rd ACM SIGCOMM 2014 Workshop on Hot Topics in Software Defined Networking, HotSDN 2014*, pp. 13–18, ACM, New York, NY, USA, August 2014.

[27] J. Rasley, B. Stephens, C. Dixon et al., "Planck: Millisecond-scale monitoring and control for commodity networks," in *Proceedings of the 2014 ACM Conference on Special Interest Group on Data Communication, SIGCOMM 2014*, vol. 44, pp. 407–418, August 2014.

[28] N. K. Sharma, A. Kaufmann, T. E. Anderson, A. Krishnamurthy, J. Nelson, and S. Peter, *Evaluating the power of flexible packet processing for network resource allocation*, pp. 67–82, 2017.

[29] D. T. Nguyen, "Modeling load uncertainty in distribution network monitoring," *IEEE Transactions on Power Systems*, vol. 30, no. 5, pp. 2321–2328, 2015.

[30] P. Peresini, M. Kuzniar, and D. Kostic, "Dynamic, fine-grained data plane monitoring with monocle," *IEEE/ACM Transactions on Networking*, vol. 26, no. 1, pp. 534–547, 2018.

[31] J. Suh, T. T. Kwon, C. Dixon, W. Felter, and J. Carter, "OpenSample: A low-latency, sampling-based measurement platform for commodity SDN," in *Proceedings of the 2014 IEEE 34th International Conference on Distributed Computing Systems, ICDCS 2014*, pp. 228–237, July 2014.

[32] M. Wang, B. Li, and Z. Li, "sFlow: towards resource-efficient and agile service federation in service overlay networks," in *Proceedings of the 24th International Conference on Distributed Computing Systems, 2004*, pp. 628–635, Tokyo, Japan, March 2004.

[33] C. Yu, C. Lumezanu, Y. Zhang, V. Singh, G. Jiang, and H. V. Madhyastha, "FlowSense: monitoring network utilization with zero measurement cost," in *Proceedings of the 14th International Conference on Passive and Active Measurement (PAM '13)*, pp. 31–41, Springer, Berlin, Germany, 2013.

[34] Z. Yang and K. L. Yeung, "An efficient flow monitoring scheme for sdn networks," in *Proceedings of the 2017 IEEE 30th Canadian Conference on Electrical and Computer Engineering (CCECE)*, pp. 1–4, April 2017.

[35] A. Nguyen-Ngoc, S. Lange, T. Zinner et al., "Performance evaluation of selective flow monitoring in the ONOS controller," in *Proceedings of the 2017 13th International Conference on Network and Service Management (CNSM)*, pp. 1–6, November 2017.

[36] P. Berde, M. Gerola, J. Hart et al., "ONOS: Towards an open, distributed SDN OS," in *Proceedings of the 3rd ACM SIGCOMM 2014 Workshop on Hot Topics in Software Defined Networking, HotSDN '14*, pp. 1–6, ACM, New York, NY, USA, August 2014.

[37] Y. Lee, S. Lee, H. Seo, C. Yoon, S. Shin, and H. Yoon, "Duo: software defined intrusion tolerant system using dual cluster," *Security and Communication Networks*, vol. 2018, Article ID 6751042, 13 pages, 2018.

[38] A. C. Oliveira, H. Chagas, M. Spohn, R. Gomes, and B. J. Duarte, "Efficient network service level agreement monitoring for cloud computing systems," in *Proceedings of the 19th IEEE Symposium on Computers and Communications, ISCC 2014*, pp. 1–6, June 2014.

[39] J. Kim, D. Kim, and S. Choi, "3GPP SA2 architecture and functions for 5G mobile communication system," *ICT Express*, vol. 3, no. 1, pp. 1–8, 2017.

[40] N. Alliance, *Ngmn 5g white paper, Next Generation Mobile Networks Ltd, Frankfurt am Main, 2015*.